

Nocturnal++ Requirements Specification

This document outlines the software requirements for the Nocturnal++ system, aligned with the provided UML class diagram architecture.

Section	Requirement	Priority	Rationale
1.1	Facade Pattern Enforcement	1	All high-level operations (login, logout, submitAnswer) must be centralized through SystemFacade to ensure a unified interface.
1.2	Singleton Enforcement	1	SystemFacade, UserList, and QuestionList must implement the Singleton pattern via getInstance() to maintain a single instance of each.
1.3	Data Management	1	SystemFacade uses DataLoader to initialize data and DataWriter to persist Users, Questions, and Answers.
2.1	Account Structure	1	Users are identified by a UUID and must store email, passwordHash, name, createdAt, and lastLogin.
2.2	Password Validation	1	The User class contains private helpers: hasUpperCase, hasLowerCase, hasDigit, and hasSpecialCharacter.
2.3	Authentication Logic	1	UserList handles core authentication via authenticate(), while SystemFacade provides the login() endpoint.
2.4	Role-Based Access	1	Privileges are determined by isAdmin and isContributor boolean flags within the User class [cite: 4, 60].
3.1	Profile Association	1	Every User has a Profile containing school, major, gradYear, and resumeURL [cite: 4, 43-47].
3.2	Profile Management	2	Profile.updateProfile() modifies academic details, and updateUpvotes() tracks engagement [cite: 4, 50].
3.3	Rank & Progress	3	Profile.getRank() returns a Rank, and User.getProgress() returns a Progress object [cite: 4, 54, 78].
4.1	Question Definition	1	InterviewQuestion includes difficulty, type, category, and an author [cite: 4, 110-114].
4.2	Question Management	1	QuestionList manages the collection using a HashMap keyed by UUID [cite: 4, 84].
4.3	Filtering & Search	1	QuestionList provides filtering via getCategory() and getByDifficulty() [cite: 4, 91-94].
4.4	Engagement Metrics	2	Questions track totalAttempts, totalSuccesses, and maintain Answers and Comments [cite: 6, 114-116].
5.1	Answer Submission	1	SystemFacade.submitAnswer() requires userID, questionID, and codeSnippet [cite: 4, 33, 34].
5.2	Interaction Logic	2	Answer and Comment classes support upvote(), downvote(), and getVoteScore() [cite: 4, 134-136, 140].
5.3	Comment Metadata	2	Comment objects track isEdited status and provide getAuthorName() [cite: 4, 127, 132].
6.1	Category Enum	1	Includes: BIG_O, ARRAY, LINKED_LIST [cite: 7, 98-100].
6.2	Difficulty Enum	1	Includes: EASY, MEDIUM, HARD [cite: 4, 103].
6.3	QuestionType Enum	1	Includes: CODING, MULTIPLE_CHOICE, SHORT_ANSWER [cite: 4, 105].